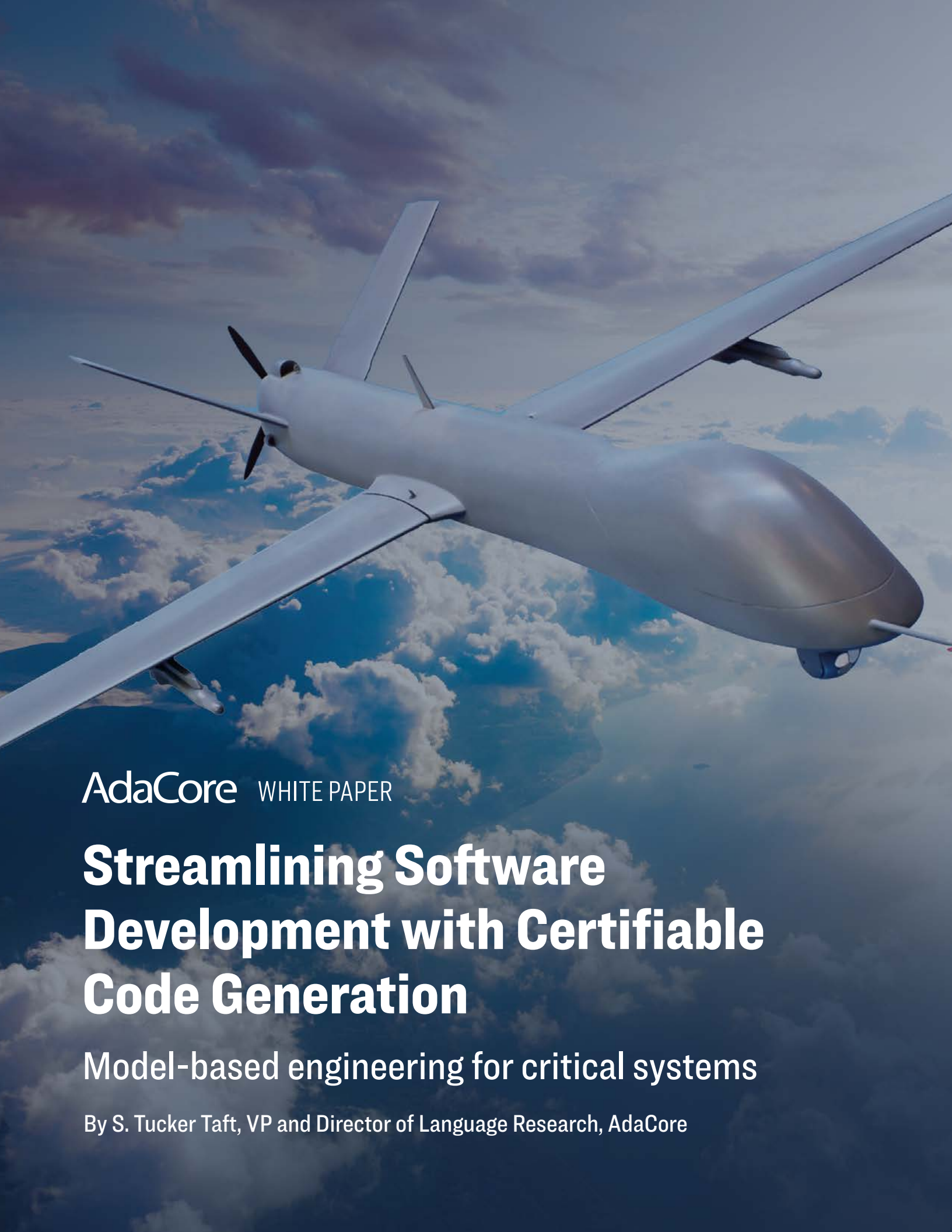

AdaCore WHITE PAPER

Streamlining Software Development with Certifiable Code Generation

Model-based engineering for critical systems

By S. Tucker Taft, VP and Director of Language Research, AdaCore

In an increasingly digital age, safety-critical software is becoming an ever more vital component of military and aerospace projects. Given its importance, any issues with writing and certifying this software can negatively impact overall project success, increasing risk and causing potential delays.

As the role of safety-critical software grows, so does its complexity and the time required to create and test code. That's why many projects now use a model-based design approach to programming. Essentially, this allows teams to create digital models of the software to be built, which can be run, checked, and refined through simulations – all before any code is written.

One of the key benefits here is that modeling languages allow domain experts, rather than software engineers, to write software specifications. As domain specialists understand more deeply what they want the safety-critical system to achieve, this ensures a closer fit to real-world requirements.

However, while this helps with the early stages of a project, often these models are then handed over to programmers to manually create code that matches the behavior of the original model. This adds time to the process and introduces the potential for programming errors that then need to be spotted and fixed later in the project, leading to potential delays.

Creating software at the touch of a button

What aerospace and defense software teams need is a way of automating the process and removing the need for manual programming. The good news is that the technology is available, with auto code generators able to take a model and automatically produce compilable source code at the touch of a button. This clearly delivers enormous time and resource savings, particularly on complex, large projects.

However, two factors have slowed the adoption of these code generators in aerospace – one internal and one external. First, off-the-shelf code generators are difficult to customize to match the internal coding standards that companies set, particularly in safety-critical systems. Due to this, our research found that that six out of 10 companies operating in the high-integrity software industry either manually translate models to source code (not using a code generator at all) or have to post-process the code produced by an existing, off-the-shelf code generator using other tools, adding to time and resource requirements.

The second issue revolves around certification. Agencies such as the U.S. Federal Aviation Administration (FAA), which are charged with protecting commercial aircraft safety, do not automatically trust this auto-generated source code. Therefore, they mandate that it goes through the same level of verification as for hand-written code, which is both extensive in terms of time, and expensive when it comes to resources. All of this adds additional complexity to projects and hinders the use of code generators.

Building trust in safety-critical code

So how can defense and aviation contractors overcome these challenges and unlock the benefits of code generators? Taking the certification issue first, to successfully deploy code generators on safety-critical projects, while significantly simplifying the process of verifying the generated code, it is essential that certification authorities have a high level of trust in the code generator. That means the tool must be approved for use on the project (a process known as tool qualification).

The DO-178C standard (and the associated tool-oriented DO-330 standard), which are required by the FAA as well as international aviation regulatory authorities, have five levels of trust, depending on the role of the tool. Referred to as “Tool Qualification Levels”, or “TQLs”, these range from the lowest, TQL-5, where a tool is just automating checking for bugs, all the way up to TQL-1 where the tool is generating flight code. In the TQL-1 case, any bugs in the tool could insert potentially catastrophic errors into the system, meaning standards are correspondingly tighter.

The benefit is that if your code generation tool achieves TQL-1 qualification, you don’t have to review and test your source code as a separate artifact – you can focus on verifying the original model and the final application. Using a TQL-1 qualified code generator is, therefore, an enormous advantage in both saving time and creating a simpler, more streamlined workflow.

However, given its criticality, achieving TQL-1 qualification is complex and rigorous, following a similar process to certification for the final software itself. Tools need to undergo detailed requirements definition and requirements-based testing. In addition, since the model itself has to be safe, a safe subset of the modeling language has to be defined to facilitate certification.

Ensuring certification across all projects

Certification requirements impact how software tools such as code generators are used and adopted within organizations. Major aerospace and defense companies are working on multiple, large, long-term software development projects, all at the same time. Clearly, to increase consistency, cost-effectiveness, and skills transfer, they will often want to use the same versions of the same tools across all projects at the same time – and these tools need to be qualified to the right TQL level.

Given that qualifying tools is an intensive process, this can mean that projects are forced to freeze on a particular, older version of a tool that has been qualified. The alternative of re-qualifying a newer version of the tool might require restarting the qualification process and/or licensing a qualification kit from the tool vendor all over again. Also, as different projects each require a qualification kit, each may have frozen on different versions of the same tools. So rather than benefiting from a single, up-to-date set of qualified tools that can be deployed across multiple projects, companies are stuck with a mishmash of solutions, many of which may lack the latest features.

QGen – A new approach

Overcoming all these issues requires a new, collaborative approach. Working with the aerospace industry, AdaCore has developed QGen, a model-based engineering toolsuite with a qualifiable code generator for the dominant Simulink® and Stateflow® modeling languages. QGen takes in Simulink/Stateflow models and outputs code either in the safety-oriented MISRA subset of C or the safe and formally verifiable SPARK subset of Ada.

AdaCore is currently in the process of qualifying the QGen code generator at the TQL-1 level. This will streamline the model-based certification process (under DO-178C and its DO-331 supplement) for users creating safety-critical systems using the tool, unlocking the benefits of code generation while ensuring that certification needs are met. Built from the ground up to be qualifiable and productive, QGen has a full suite of model-based engineering tools that can expedite the certification/testing process (including testing tools and a model level debugger), providing everything required in a single package.

Importantly, QGen eliminates any concerns around having to qualify tools across multiple projects. The TQL-1 Enterprise Qualification Package saves money by providing a single, company-wide license to cover all projects within an organization – and enables the sharing of the same qualification kit across all projects. There's no need to freeze on older versions of tools. Customers also benefit from an annual update based on their individual needs – so they can add a second generated source language, support another certification authority (such as EASA, the European equivalent of the FAA) or move to a newer version of Simulink, all under the same agreement. All projects benefit from the same warranties, including expert assistance for certification audits, as well as in-depth technical support from the team that created the tool.

Essentially, QGen is designed to combine the benefits of automatic code generation with more straightforward certification, with a solution developed specifically for the needs of large aerospace and defense organizations running multiple, simultaneous projects. This means that customers who want to take a model-based approach to development can now dramatically reduce verification activities on the generated code, lowering costs while streamlining the overall certification process.

Certified code generation in action

Collins Aerospace selected QGen, and the TQL-1 Enterprise Qualification Package, to advance the development of its FAA-certifiable Perigon™ computer, which is designed to support the future flight control and vehicle management needs of commercial and military rotary/fixed-wing platforms. By using the TQL-1 release of QGen, Perigon™ software developers are able to save thousands of hours of testing, verification, and certification efforts, while providing additional safety guarantees to their customers. With the adoption of the QGen Enterprise Qualification Package, Collins is now able to streamline its model-based engineering practices.

MHI Aerospace Systems Corporation (MASC), a member of the Mitsubishi Heavy Industries Group, selected QGen to develop the software for the Throttle Quadrant Assembly (TQA) system. This avionics research project is being conducted to meet the Level C objectives in the DO-178C safety standard for airborne software and its DO-331 supplement on Model-Based Development and Verification.

"We chose QGen as our Auto Code Generator because we believe TQL-1 qualification will reduce the effort of our verification activities that comply with DO-331," said Hiroyuki Kakamu, General Manager of MASC. "As part of the TQA project, we discussed DO-331 compliance with AdaCore's engineers. Based on these discussions, we developed a Model-Based Design process that complies with DO-331 based on the use of QGen."

The importance of software to safety-critical projects is growing – companies today need to manage more code and greater complexity while hitting tighter project deadlines. While model-based software development combined with auto code generation delivers the required time and resource savings, certification and qualification have previously been a significant overhead on projects involving code generators, reducing these benefits. QGen unlocks the value in this approach, providing a comprehensive solution that combines tools and support to enable the more streamlined certification of automatically generated source code. This reduces time and risk, helping hit key milestones and contributing to overall project success.

Find out how AdaCore can help you accelerate your safety-critical software development. Contact sales@adacore.com

---

S. Tucker Taft
VP and Director of Language Research
AdaCore

S. Tucker Taft is VP and Director of Language Research at AdaCore and a Senior Advisor for AdaCore's model-based development technology, QGen. Mr. Taft joined AdaCore in 2011 as part of a merger with SofCheck, which he had founded in 2002 to develop advanced static analysis technology. Prior to that Mr. Taft was a Chief Scientist at Intermetrics, Inc. and its follow-ons for 22 years, where in 1990-1995 he led the design of Ada 95. Mr. Taft received an A.B. Summa Cum Laude degree from Harvard University, where he has more recently taught compiler construction and programming language design.